Formalization of Polynomially Bounded and Negligible Functions Using the Computer-Aided Proof-Checking System Mizar

Hiroyuki Okazaki and Yuich Futa

Graduate School of Science and Technology,Shinshu University
4–17–1 Wakasato Nagano–city, Nagano 380–8553, Japan
Japan Advanced Institute of Science and Technology
okazaki@cs.shinshu-u.ac.jp

Abstract. In recent years, formal verification applications have attracted significant attention. In particular, verification of the security of cryptosystems has been investigated extensively. In this study, we attempt to develop various mathematical libraries for cryptology using the Mizar proof checking system. Polynomially bounded and negligible functions play very important roles in cryptology. Therefore, we introduce formalized definitions of polynomially bounded and negligible functions for formalizing cryptology in Mizar.

Keywords: polynomially bounded functions, negligible functions

1 Introduction

In recent years, formal verification applications have attracted significant attention. Verification of the security of cryptosystems is has been investigated extensively. The objective of this study is to develop mathematical libraries to verify the security of cryptographic systems using the Mizar proof checking system [1]. Mizar is one of the best-known projects in formalized mathematical knowledge management. To achieve our objective, we attempt to formalize various topics related to cryptology, such as computational complexity, probability and probability distributions [2–7], algorithms [8], and number theory [9, 10]. The security of most current public-key cryptosystems is based on computational complexity. A public-key cryptosystem is secure if attacks against the cryptosystem are not easier than solving a problem for which there is no known efficient algorithm. Discrete-logarithm and factorization problems are considered to be hard to solve and are therefore employed in public-key cryptosystems [11, 12]. The difficulty of such problems is classified according to their computational time.

Discrete-logarithm and factorization are subexponential-time problems that are difficult to solve if the input size is sufficiently large. It is well known that polynomial-time problems can be solved with a relatively low computational cost. We consider a problem P_a to be as hard to solve as another problem P_b if we can solve P_b by calling the algorithm that can solve P_a a polynomially

bounded number of times. We are interested in the formalization of computational complexities. Polynomially bounded functions play an important role in practical computational complexity theory. The class P is a fundamental computational complexity class that contains all polynomial-time decision problems [13]. It takes a polynomially bounded amount of computation time to solve polynomial-time decision problems using a deterministic Turing machine. Therefore, we formalize a rigorous definition of polynomially bounded functions.

To analyze the security of cryptographic primitives such as symmetric encryption or hash functions as well as that of cryptographic protocols, we evaluate the success probability of any attack against them. In general, a cryptosystem is secure if the probability of any attack succeeding against the cryptosystem is negligible. Therefore, we formalize a rigorous definition of the negligible functions [14]. Both polynomially bounded and negligible functions are described by the behavior of polynomials. Polynomially bounded functions do not increase faster than polynomials themselves, and negligible functions do not decrease slower than the multiplicative inverse of polynomials. There are interesting relationships between polynomially bounded and negligible functions.

In this paper, we first introduce our formalized definition of polynomially bounded functions. Next, we formalize the definition of negligible functions and prove that the set of negligible functions is a subset of the set of polynomially bounded functions. Moreover, we introduce an equivalence relationship between polynomially bounded functions using negligible functions. All formalized definitions and theorems in this paper are described in the Mizar language, and their proofs are verified by the Mizar proof checker. Our formal descriptions have been stored in the current version of the Mizar Mathematical Library (MML)* and are available online on the Mizar Project official website [1].

1.1 Our contribution

Asymptotic notations allow us to describe the behavior of given real-valued functions. In particular, O notation is often used to analyze the increase in real-valued functions and to classify the computational complexity of solving the given problems. Fortunately, there are formal definitions and theorems regarding asymptotic notation in the MML. Therefore, we can use these definitions to offer a formalized definition of polynomially bounded functions that do not increase faster than polynomials. We then formalize some theorems regarding polynomially bounded functions. Next, we formalize the definition of negligible functions that do not decrease slower than the multiplicative inverses of polynomials and formalize some theorems about them. It is well known that linear combinations of polynomially bounded functions are also polynomially bounded and that those of negligible functions are also negligible. These facts are very useful for evaluating computational complexity and for proving the security of cryptosystems. Thus, we formalize proofs for these theorems. Specifically, we

* Indexed by the MML identifiers ASYMPT_2[15] and ASYMPT_3[16], for Mizar version: 8.1.04 5.32.1246 or later.

formalize the algebraic structure of polynomially bounded functions. We then introduce an equivalence between the elements of the algebra of polynomially bounded functions using negligible functions.

2 Mizar

Mizar [1] is one of the best-known projects in formalized mathematical knowledge management. The Mizar system comprises the Mizar language, an automated proof verifier, and mathematical libraries. The Mizar language normally uses first-order predicate logic; however, it can use second-order predicate logic with conditions.

3 Preparation

3.1 Asymptotic notation

In this section, we briefly review the asymptotic $\mathcal{O}$ notation and introduce related formal definitions available in the MML. For our purposes (i.e., computer science and cryptology) it is sufficient to analyze the behavior of discrete functions. In fact, asymptotic notation in the MML is formalized on sequences, which are synonymous with functions whose domains comprise natural numbers.

Definition: 1 *Let $f(\cdot)$ and $g(\cdot)$ be functions from $\mathbb{N}$ to $\mathbb{R}$. $g(\cdot) \in \mathcal{O}(f(\cdot))$ iff $\exists N$ is a natural number and c is a real number s.t., $\forall n$ s.t. $N \leq n$ holds $0 \leq g(n) \leq c \cdot f(n)$.*

The $\mathcal{O}$ notation is defined in Mizar as follows:

Definition: 2 `(ASYMPT_0:def 9)`

```
definition
let f be
    eventually-nonnegative Real_Sequence;
func Big_Oh(f) -> FUNCTION_DOMAIN of NAT,REAL
 equals
{ t where t is Element of Funcs(NAT, REAL)
  : ex c,N st c > 0 &
  for n st n >= N holds t.n <= c*f.n &
  t.n >= 0};
end;
```

In Mizar, Real_Sequence is an alias for a function from $\mathbb{N}$ to $\mathbb{R}$ and "eventually-nonnegative" is an attribute for Real_Sequence and is defined as follows:

Definition: 3 `(ASYMPT_0:def 2)`

```
definition
  let f be Real_Sequence;
  attr f is eventually-nonnegative means
```

```
  ex N being Nat st for n being Nat
  st n >= N holds f.n >= 0;
end;
```

The sequence of the monomial $f(n) = n^a$ is defined in Mizar as follows:

Definition: 4 (ASYMPT_1:def 3)
```
definition
  let a be Real;
  func seq_n^(a) -> Real_Sequence means
  it.0 = 0 &
  for n st n > 0 holds
    it.n = n to_power a;
end;
```

The sequence of the exponential $a^{b \cdot n+c}$, where a, b, and c are given constant real numbers, is defined in Mizar as follows:

Definition: 5 (ASYMPT_1:def 1)
```
definition
  let a,b,c be Real;
  func seq_a^(a,b,c) -> Real_Sequence means
  it.n = a to_power (b*n+c);
end;
```

Note that for a given real number x, $f(n) = x^n$ is represented as "seq_a^(x,1,0)" in Mizar.

3.2 Polynomially bounded functions

In this section, we briefly review polynomially bounded functions.

Definition: 6 *A real valued function, $f(\cdot)$, is polynomially bounded if there exists a natural number, k, such that*

$$f(x) \in \mathcal{O}\left(x^k\right)$$

3.3 Negligible functions

In this section, we briefly review negligible functions[14].

Definition: 7 *Let $\mu(\cdot)$ be a function from $\mathbb{N}$ to $\mathbb{R}$. $\mu(\cdot)$ is a* (negligible function) *iff for all polynomial $p(\cdot)$ holds $\exists N$ be a natural number s.t., $\forall n$ be a natural number s.t. $N \leq n$ holds*

$$\mu(n) < \frac{1}{|p(n)|}$$

4 Formalization of polynomially bounded functions and polynomial functions

In this section, we introduce our formal definitions of polynomially bounded functions and polynomial functions.

4.1 Formalized definition of polynomially bounded functions

We propose a formal definition of polynomially bounded functions as follows:

Definition: 8 (ASYMPT_2:def 1)
```
definition
  let p be Real_Sequence;
  attr p is polynomially-bounded means
  ex k be Nat st
    p in Big_Oh(seq_n^(k));
end;
```

This definition may appear very similar to Def. 6 but is only a formalization of an attribute for real sequences. In the Mizar language, attributes are constructors of adjectives. We can define an attribute followed by the keyword "attr". Def. 6 gives only a definition of an attribute for real-valued functions. However, formal methods never accept the declaration of new objects without an evidence of their existence in order to eliminate any errors in proofs. In the Mizar system, there are several ways to treat polynomially bounded functions. One is to prove that declared real sequences satisfy the definition of being polynomially bounded. Another way is to prove the existence of polynomially bonded functions and to define a new mode for polynomially bounded sequences as a sub mode of a "Real_Sequence". It should be noted that in the Mizar language, "mode" is similar to "type" used in other languages. Furthermore, all modes are derived from the mode "set" because the Mizar system is based on the set theory. The third option is to define a set of specific functions and their algebraic structure as new terms. We will show an example of the third case in Sec. 6.

Let us now introduce some theorems about polynomially bounded functions.

Theorem: 1 (ASYMPT_2:25)
```
theorem
  for a be Nat st 1 < a holds
  seq_a^(a,1,0) is non polynomially-bounded;
```

This theorem shows that the exponential $f(x) = a^x$ is not polynomially bounded if $1 < a$. We first prove Theorem 2 as a lemma for proving Theorem 1. The following theorem holds that 2^x is nonpolynomially bounded:

Theorem: 2 (ASYMPT_2:16)
```
theorem
  for x be Nat st 1 < x holds
```

```
  not ex N,c be Nat st
  for n be Nat st N <= n holds
  2 to_power n <= c * (n to_power x);
```

To prove Theorem 2, we assume that the proposition of Theorem 1 is not true and derive a contradiction against Theorem 2. The actual formal proof of Theorem 1 encoded in the Mizar language is described in [15].

4.2 Formalization of polynomial functions

Although polynomials have been formalized in the MML [17], we propose a formal definition of univariate polynomials as sequences for formalizing negligible functions. Because this formalization defines polynomials as an algebraic structure of multivariate polynomials, it is directly unsuitable for asymptotic notation.

Definition: 9 (ASYMPT_2:def 2)

```
definition
  let c be XFinSequence of REAL;
  func seq_p(c) -> Real_Sequence
  means
  for x be Nat holds
  it.x = Sum(c (#) seq_a^(x,1,0));
end;
```

"XFinSequence" is a finite sequence whose indices start from 0. In this definition, XFinSequence c is the given coefficient for the representation of a polynomial. For example, (seq_p(<*a,b,c*>)).x = a + bx + cx^2. Note that this definition formalizes "func seq_p(c) ", where "func" is a keyword that shows that the term is defined as a FUNCTOR. In the Mizar language, FUNCTOR plays the role of a constructor of new terms. To define a FUNCTOR, the proof checker requires proofs of at least the existence and uniqueness of the new term. In this case, once we call "seq_p(c)", the checker automatically recognizes this term as a Real_Sequence. We can then refer to the defining theorem, "for x be Nat holds it.x = Sum(c (#) seq_a^(x,1,0))" to prove another theorem. We will discuss the usage of FUNCTOR in Sec. 6.

Let us now introduce theorems about polynomials.

Theorem: 3 (ASYMPT_2:45)

```
theorem
  for k be Nat, c be XFinSequence of REAL
  st len c = k+1 & 0 < c.k
  holds seq_p(c) in Big_Oh( seq_n^(k) );
```

This theorem holds that any kth-degree polynomial function whose most significant coefficient is positive is of $\mathcal{O}(n^k)$. Theorem 4 follows Theorem 3:

Theorem: 4 (ASYMPT_2:46)
```
theorem
  for k be Nat, c be XFinSequence of REAL
  st len c = k+1 & 0 < c.k holds
   seq_p(c) is polynomially-bounded;
```

This theorem holds that all univariate polynomials are polynomially bounded.

5 Formalization of negligible functions

In this section, we introduce our formal definition of a negligible function.

Definition: 10 (ASYMPT_3:def 4)
```
definition
 let f be Function of NAT,REAL;
  attr f is negligible means
  for c be non empty positive-yielding
      XFinSequence of REAL holds
  ex N be Nat st
  for x be Nat st N <=x holds
  |. f.x .| < 1/((seq_p(c)).x);
end;
```

We then introduce some theorems about negligible functions. The following theorem holds that 2^{-x} is a negligible function:

Theorem: 5 (ASYMPT_3:25)
```
theorem
for  f be Function of NAT,REAL
  st for x be Nat holds
    f.x = 1/ (2 to_power x)   holds
  f is negligible;
```

Theorem 6 holds that negligible functions converge to zero.

Theorem: 6 (ASYMPT_3:24)
```
theorem
  for f be Function of NAT,REAL
  st f is negligible
  holds f is convergent & lim f = 0;
```

By providing a formal proof of correctness, we can automate inference rules (i.e., namely clusters) in the MML as follows:

```
registration
  let f be negligible Function of NAT,REAL,
  a be Real;
  cluster a(#)f -> negligible
```

```
                  for Function of NAT,REAL;
end;
registration
  let f,g be negligible Function of NAT,REAL;
  cluster f+g -> negligible
                   for Function of NAT,REAL;
end;
registration
  let f,g be negligible Function of NAT,REAL;
  cluster f(#)g -> negligible
                   for Function of NAT,REAL;
end;
registration
  let f be negligible Function of NAT,REAL,
  g be polynomially-abs-bounded
                      Function of NAT,REAL;
  cluster g(#)f -> negligible
                   for Function of NAT,REAL;
end
```

Once these clusters are registered, the checker automatically infers that addition and scalar multiplication between negligible functions yield negligible results. For example, the Mizar proof checker inferences the following theorem automatically:

Theorem: 7

```
for a be Real,
e,f be negligible Real_Sequence,
 g be polynomially-abs-bounded
    Real_Sequence holds
  (a (#) f)(#) g + e is negligible;
```

In general, a cryptosystem is secure if the probability of a successful attack against the cryptosystem is negligible. To prove this, we evaluate the total sum of the probability of success of any possible attack. Adversaries are allowed to attack in parallel or iteratively. In most cases, it is sufficient to evaluate the linear combination of probabilities of successful attacks if the probability of all attacks is negligible. We have formalized definitions and theorems concerning probability and probability distributions that can treat concrete probabilistic events for describing attacks against cryptosystems [2–7]. In combination with these formal descriptions, the above-mentioned clusters would be useful for formal verification of the security of cryptosystems.

6 Formalization of algebra of polynomially bounded functions

In Secs. 4 and 5, we introduced our formalizations of polynomially bounded functions and negligible functions. In this section, we propose a formal definition

of the algebra of polynomially bounded functions and present some theorems about negligible functions and this algebra.

6.1 Algebra of polynomially bounded functions

Set of polynomially bounded functions First, we define a relaxed attribute of polynomially bounded functions namely "polynomially-abs-bounded", as follows:

Definition: 11 (ASYMPT_3:def 1)
```
definition
  let p be Real_Sequence;
 attr p is polynomially-abs-bounded means
 ex k be Nat st |. p .| in Big_Oh(seq_n^(k));
end;
```

Here, |. p .|is the absolute value of the given function p. Although we define the attribute polynomially-bounded in Sec. 8, we formalize this definition for technical reasons, i.e., in order to introduce the multiplicative inverse of addition between polynomially bounded functions. Naturally, polynomially bounded functions are polynomially-abs-bounded. We then register the following clusters:

```
registration
  cluster polynomially-bounded ->
  polynomially-abs-bounded for Real_Sequence;
end;
```

Next, we define the set of polynomially-abs-bounded functions as a new term, namely, "Big_Oh_poly", as follows:

Definition: 12 (ASYMPT_3:def 2)
```
definition
func Big_Oh_poly -> Subset of RAlgebra (NAT)
    means  for x being object holds x in it
    iff  x is polynomially-abs-bounded
         Function of NAT,REAL;
end;
```

We then register the following cluster:

```
registration
  cluster Big_Oh_poly -> non empty;
end;
```

In other words, we made the proof-checker recognize the existence of polynomially-abs-bounded functions.

Algebraic structure of polynomially bounded functions In this section, we define the algebraic structure of polynomially bounded functions as follows:

Definition: 13 (ASYMPT_3:def 3)
```
definition
  func R_Algebra_of_Big_Oh_poly
             -> strict AlgebraStr means
  the carrier of it = Big_Oh_poly
  & the multF of it
      = (RealFuncMult(NAT)) || Big_Oh_poly
  & the addF of it
      = (RealFuncAdd(NAT)) || Big_Oh_poly
  & the Mult of it  = (RealFuncExtMult(NAT))
                       | [:REAL,Big_Oh_poly:]
  & the OneF of it
      = RealFuncUnit(NAT)
  & the ZeroF of it
      = RealFuncZero(NAT);
end;
```

In this formalization, we define the carrier, multiplication, addition, scalar multiplication, multiplicative unit, and additive unit. The carrier of R_Algebra_of_Big_Oh_poly (i.e., Big_Oh_poly) is included in the set of all real-valued functions. Instead of defining a new operation, we use predefined operations on real-valued functions by restricting their domain according to the carrier of this structure. We define the additive and multiplicative units of this structure as being the same as those of the algebra of real-valued functions.

We then prove the attributes that are held by the algebraic structure R_Algebra_of_Big_Oh_poly, and register them as the following cluster:

```
registration
  cluster R_Algebra_of_Big_Oh_poly
  -> strict Abelian add-associative
   right_zeroed right_complementable
   commutative associative right_unital
   right-distributive  vector-associative
   scalar-associative vector-distributive
   scalar-distributive;
end;
```

As a result, we proved the following theorem:

Theorem: 8 (ASYMPT_3:18)
```
theorem
  R_Algebra_of_Big_Oh_poly is Algebra;
```

6.2 Set of negligible functions

In this section, we define the set of negligible functions as a new term, namely, "negligibleFuncs", as follows:

Definition: 14 (ASYMPT_3:def 5)

```
definition
 func negligibleFuncs ->
        Subset of Big_Oh_poly
    means
 for x being object holds x in it
 iff x is negligible Function of NAT,REAL;
end;
```

We then register the following cluster:

```
registration
  cluster negligibleFuncs -> non empty;
end;
```

Note that we define "negligibleFuncs" as a nonempty subset of R_Algebra_of_Big_Oh_poly.

In the rest of this section, we describe particularly useful theorems about polynomially bounded functions and negligible functions.

Equivalence between polynomially bounded functions We define the following predicate:

Definition: 15 (ASYMPT_3:def 6)

```
definition
  let f,g be Function of NAT,REAL;
  pred f negligibleEQ g means
  ex h be Function of NAT,REAL st
   h is negligible &
  for x be Nat holds
     |. f.x - g.x .| <= |.h.x.|;
  reflexivity;
  symmetry;
end;
```

Here "f negligibleEQ g" means that there exists a negligible function, h, which is the upper bound of the difference between two functions, f and g. In other words, we introduce an equivalence relationship between polynomially bounded functions. This equivalence would be useful not only for cryptography and computer science but also for approximating and analyzing functions.

Moreover, we prove the following theorem about multiplication between polynomially bounded and negligible functions:

Theorem: 9 (ASYMPT_3:39)

```
theorem
  for v,w be
   VECTOR of R_Algebra_of_Big_Oh_poly
   st w in negligibleFuncs
   holds v * w in negligibleFuncs;
```

where VECTOR is a synonym for an element of the structure, and "v * w" is the product of two VECTORs v and w: $(v * w).x = (v.x) * (w.x)$. This theorem shows that the product "v * w" of two functions is negligible if v is a polynomially bounded function and w is a negligible function. Recall the above mentioned theorems and clusters concerning negligible functions. The set of negligible functions is a nonempty subset of the set of polynomially bounded functions. Addition and scalar multiplication between negligible functions also yield negligible functions. These theorems mean that the set of negligible functions is an ideal of the set of polynomially bounded functions.

7 Case study: Evaluating the computational cost of algorithms

In this section, we show an example of evaluating the computational cost of algorithms. In [8], we did not formalize about computational cost of the Euclidean algorithm, although we proved its correctness. Therefore we formalize Lamé's theorem to prove that the Euclidean algorithm is a polynomial-time algorithm. The Euclidean algorithm is an efficient algorithm that computes the greatest common divisor (GCD) of two given integers. Lamé's theorem [18] is known as the first application of Fibonacci numbers and it shows that the Euclidean algorithm terminates after a maximum of $5X$ repetitions, where X represents the decimal digits of the smaller one of the two given integers. The Euclidean algorithm terminates after n steps such that $\mathrm{Fib}(n-1) \leq b$, where b is the smaller of the two given integers. By evaluating Fibonacci numbers, it was found that n is not larger than $5X$ repetitions, where X represents the decimal digits of b. Consequently, we conclude that n is not increasing faster than the polynomials. In other words, we can evaluate that the computational cost of the Euclidean algorithm is being polynomially bounded.

Theorem: 10

```
theorem
for a,b be Element of INT st
  |.a.| > |.b.| & b > 1 holds
  ex A,B be sequence of NAT,
    C be Real_Sequence,
     n be Element of NAT st
  A.0 = |.a.| & B.0 = |.b.| &
  (for i be Nat holds
  A.(i+1) = B.i &
```

```
  B.(i+1) = A.i mod B.i) &
  n = (min*{i where i is Nat: B.i = 0} ) &
a  gcd b  = A.n &
Fib(n+1) <= |. b .| &
n <= 5*[/ log(10,|. b .|) \] &
    n <= C.(|. b .|) & C is polynomially-bounded;
```

In future works, we will formalize complex algorithms that include iterations or subroutine calls even though we were able to formalize the Euclidean algorithm using simple sequences.

8 Overview of verifying security of cryptologic systems

In this section, we briefly describe our future plans for formalizing the security of cryptologic systems. It is well known that the security of cryptosystems is dependent on the complexity of computational problems. Polynomially bounded functions play an essential role in evaluating and classifying the complexity of computational problems. We are attempting to formalize computational complexity using our libraries mentioned in Sec. 4.

Probability distribution is a fundamental principal of cryptology. We can define other cryptologic topics using our formalization of probability distribution. Currently, we are attempting to formalize the indistinguishability of probability distributions. In modern cryptology, the security of cryptologic systems is essentially defined by indistinguishability. For example, various cryptologic topics, such as pseudo-random number generators and hash functions, are described using the concept of indistinguishability. We often design a cryptographic scheme by employing ideal functions. However, we must replace such ideal functions with feasible functions when we implement the schemes. Thus, an implemented scheme is not always secure even if its ideal functionality has been proven to be secure in design. Indistinguishability of functions means the indistinguishability of the distributions of the outputs of the functions between an ideal function (e.g., an ideal random function) and a feasible function (e.g., a hash function) to prove the security of a cryptographic scheme. Two probability distributions are statistically indistinguishable if the distance between them is negligible. We intend to define a formalization of indistinguishability using our formalization of negligible functions mentioned in Sec. 5 and probability distributions[2].

9 Conclusion

In this paper, we introduced a formalized definition of polynomially bounded and negligible functions in the Mizar language. We then showed formalized theorems related to these definitions. All formalized definitions and theorems in this paper are described in the Mizar language, and the proofs of these theorems have been verified using the Mizar proof checker. Our formal descriptions have been stored in the current version of the MML and are available online on the Mizar Project

official website. We prepared these formalized articles in order to achieve our aim of constructing an automated formal verification tool for cryptology. We will now attempt to formalize one of the most important concepts of cryptology indistinguishability using the formalization of negligible functions introduced in this paper.

ACKNOWLEDGEMENT

This study is partly supported by JSPS KAKENHI 15K00183. The authors would like to express their gratitude to Prof. Yasunari Shidama for his support and encouragement.

References

1. Mizar System: *Available at* http://mizar.org/.
2. H. Okazaki, Y. Futa, Y. Shidama, "Formal definition of probability on finite and discrete sample space for proving security of cryptographic systems using Mizar", Artificial Intelligence Research, 2(4), pp.37-48, 2013. DOI : 10.5430/air.v2n4p37
3. H. Okazaki, Y. Shidama, "Random Variables and Product of Probability Spaces", Formalized Mathematics, 21(1), pp.33-39, 2013. DOI : 10.2478/forma-2013-0003
4. H. Okazaki, "Posterior Probability on Finite Set", Formalized Mathematics, 20(4), pp.257-263, 2013. DOI : 10.2478/v10037-012-0030-0
5. H. Okazaki, Y. Shidama, "Probability Measure on Discrete Spaces and Algebra of Real Valued Random Variables", Formalized Mathematics, 18(4), pp.213-217, 2010. DOI : 10.2478/v10037-010-0026-6
6. H. Okazaki, "Probability on Finite and Discrete Set and Uniform Distribution" , Formalized Mathematics, 17(2), pp.173-178, 2009. DOI : 10.2478/v10037-009-0020-z
7. H. Okazaki, Y. Shidama, "Probability on Finite Set and Real-Valued Random Variables", Formalized Mathematics, 17(2), pp.129-136, 2009. DOI : 10.2478/v10037-009-0014-x
8. H. Okazaki, Y. Aoki, Y. Shidama, "Extended Euclidean Algorithm and CRT Algorithm", Formalized Mathematics 20(2), pp-175-179, 2012. DOI : 10.2478/v10037-012-0020-2
9. Y. Futa, D. Mizushima, H. Okazaki, " Formalization of Gaussian integers, Gaussian rational numbers, and their algebraic structures with Mizar", ISITA 2012: pp.591-595, 2012.
10. Y. Futa, H. Okazaki, Y. Shidama, "Formalization of Definitions and Theorems Related to an Elliptic Curve Over a Finite Prime Field by Using Mizar", J. Autom. Reasoning 50(2), pp.161-172 (2013). DOI : 10.1007/s10817-012-9265-2
11. Ronald L. Rivest, Adi Shamir, Len M. Adelman, "A Method for Obtaining Digital Signature and Public-key Cryptsystems", MIT-LCS-TM-082, (1977).
12. Taher Elgamal, "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, IT-31(4), pp. 469?-472, (1985).
13. Jon Kleinberg, Eva Tardos, "Algorithm Design", Addison-Wesley, 2005.
14. Goldreich, "Foundations of Cryptography : Volume: 1 Basic Tools", Cambridge University Press, 2001.

15. H. Okazaki, Y. Futa, "Polynomially Bounded Sequences and Polynomial Sequences", Formalized Mathematics, 23(3), pp.205-213, 2015. DOI: 10.1515/forma-2015-0017
16. H. Okazaki, "Algebra of Polynomially Bounded Sequences and Negligible Functions", Formalized Mathematics, 23(4), pp.371-378, 2015.
17. P. Rudnicki, A. Trybulec, "Multivariate Polynomials with Arbitrary Number of Variables", Formalized Mathematics, 9(1), pp. 95–110, 2001.
18. G. Lamé', "Note sur la limite du nombre des divisions dans la recherche du plus grand commun diviseur entre deux nombres entiers", Comptes Rendus Acad. Sci, 19, pp.867–870, 1844.